Cálculo de Autovalores e Autovetores

Prof. Alexandre Salles da Cunha e Profa. Ana Paula Couto

UFMG
UNIVERSIDADE FEDERAL
DE MINAS GERAIS

UFMG - ICEx
DEPARTAMENTO DE CIÊNCIA DA
COMPUTAÇÃO

1. Algoritmos para fatoração espectral de matrizes reais simétricas (ou Hermitianas) ($S = Q^* \Lambda Q$).

2. Muito brevemente, vamos também discutir aplicações/modificações destes algoritmos para o caso não simétrico ($A = QTQ^*$).

3. Algoritmos para cálculo de valores e vetores singulares ($A = U\Sigma V^T$).

- Resolver o polinômio característico em λ, obtido ao se impor

$$det(A - \lambda I) = 0,$$

não é uma opção computacionalmente viável: o problema de obter raízes de polinômios é bastante mal condicionado.

- As técnicas empregadas para o cálculo de autovalores baseiam-se em decomposições ou fatorações de A, que revelem os autovalores.

- Por exemplo, algumas matrizes quadradas admitem decomposição espectral, na forma:

$$A = X\Lambda X^{-1} \iff AX = X\Lambda$$

onde as colunas de X fornecem os n autovetores linearmente independentes de A e Λ é uma matriz diagonal com seus autovalores.

- A transformação $A = X\Lambda X^{-1}$ é dita *similar*. Os autovalores de Λ são os autovalores de A.

- As matrizes A que admitem uma decomposição espectral $A = X\Lambda X^{-1}$ são chamadas diagonalizáveis.

$\Rightarrow$ Uma matriz é diagonalizável se e somente se não é defectiva, isto é, não possui autovalores defectivos (um autovalor é defectivo se sua multiplicidade algébrica e geométrica são distintas).

- Nem todas as matrizes são diagonalizáveis.

- Algumas matrizes não apenas admitem uma diagonalização $A = X\Lambda X^{-1}$, mas também admitem uma diagonalização unitária, isto é:

$$A = Q\Lambda Q^*,$$

onde a matriz Q (cuja conjugada transposta é Q^*) é unitária.

- Veja que em $A = Q\Lambda Q^*$ os autovetores de A além de serem linearmente independentes, foram ortogonalizados.

- A menos das entradas em Λ (que podem ter sinais negativos e eventualmente complexos), a fatoração $A = Q\Lambda Q^*$ é tanto uma fatoração espectral quanto uma decomposição em valores singulares de A.

$\Rightarrow$ Uma matriz admite uma diagonalização unitária se e somente se a matriz é normal, isto é, $A^*A = AA^*$.

Uma das fatorações mais importantes em Computação Científica é a fatoração de Schur

$$A = QTQ^*$$

onde Q é unitária e T é triangular superior.

- Toda matriz quadrada admite uma fatoração (ou triangularização) de Schur, inclusive as matrizes defectivas.
- Veja que como A e T são similares, os autovalores de A são os autovalores de T, que estão localizados na diagonal da triangular superior T.

1. Diagonalização de $A = X\Lambda X^{-1}$ existe se e somente se A é não defectiva.

2. Diagonalização unitária de $A = Q\Lambda Q^{-1}$ existe se e somente se A é normal.

3. Triangularização unitária (Decomposição de Schur) de $A = QTQ^*$ sempre existe.

Escopo dos algoritmos discutidos aqui

- Os algoritmos que apresentamos se aplicam ao caso em que A é simétrica (ou Hermitiana) e admite uma diagonalização unitária ou não, fornecendo uma fatoração de Schur.
- Vamos tratar mais o caso simétrico e fazer observações sobre qual tipo de informações tais algoritmos poderiam revelar para o caso não simétrico.

1. No caso geral (A não é simétrica ou hermitiana), devemos fazer uma fatoração de Schur, dado que toda matriz admite uma triangularização unitária.

$\Rightarrow$ Se A for normal, a matriz T obtida na fatoração de Schur será diagonal.

2. Em particular, se A for hermitiana, podemos tirar proveito desta "simetria"ao longo da computação, reduzindo os cálculos à metade, para diagonalizar A.

3. Os algoritmos que implementam estas fatorações são estáveis, pois as transformações envolvem matrizes unitárias, que preservam $\|\cdot\|_2$.

Método da Potência

Escolha um $x \in \mathbb{R}^n \setminus \{0\}$, e calcule $\frac{Ax}{\|Ax\|}, \frac{A^2x}{\|A^2x\|}, \ldots, \frac{A^kx}{\|A^kx\|}$. Quando $k \to \infty$, $\frac{A^kx}{\|A^kx\|}$ converge para um autovetor de A, associado ao maior autovalor em módulo.

- Ideia simples, mas pouco efetiva.
- Exceto para matrizes muito particulares, converge lentamente.
- Quando $\lambda_1 \approx \lambda_2$ (maior e segundo maior autovalores) são próximos, converge muito lentamente.

```
function [Ax,lambda] = MetodoPotencia(A,tol)
  k = 0;
  [m,n] = size(A)
  Ax = ones(m,1)/norm(ones(m,1),2)
  convergiu = 0;
  while (convergiu == 0)
     xantes = Ax
     Ax = A * xantes
     lambda = norm(Ax,2)
     Ax = Ax / lambda
     s = norm(Ax-xantes,2)
     if (norm(s,2) < tol)
         convergiu = 1
     end
     printf("k = %4d Norma diferenca: %7.6E \n",k,s)
     k = k + 1
  end
  Ax = Ax/norm(Ax,'inf')
endfunction
```

```
A  =

   9.    1.
   1.    2.

-->[eigenvector,eigenvalue] = MetodoPotencia(A,1.0E-6)
k =    0 Norma diferenca: 4.889354E-01
k =    1 Norma diferenca: 1.210077E-01
k =    2 Norma diferenca: 2.486851E-02
k =    3 Norma diferenca: 5.062677E-03
k =    4 Norma diferenca: 1.030241E-03
k =    5 Norma diferenca: 2.096479E-04
k =    6 Norma diferenca: 4.266207E-05
k =    7 Norma diferenca: 8.681469E-06
k =    8 Norma diferenca: 1.766626E-06
k =    9 Norma diferenca: 3.594975E-07
 eigenvector  =

   1.
   0.140055
 eigenvalue  =

   9.1400549
```

$\Rightarrow$ Sabemos que encontrar os autovalores de A equivale ao problema de encontrar as raízes de seu polinômio característico

$$det(A - \lambda I) = 0.$$

$\Leftarrow$ Por outro lado, qualquer problema de encontrar as raízes de um polinômio pode ser formulado como o problema de encontrar os autovalores de uma matriz associada ao polinômio.

Vamos verificar este resultado a seguir.

Considere o polinômio de grau m em z:

$$p(z) = z^m + a_{m-1}z^{m-1} + \cdots + a_1 z + a_0$$

e a matriz A de ordem m:

$$A = \begin{bmatrix} 0 & & & & & -a_0 \\ 1 & 0 & & & & -a_1 \\ & 1 & 0 & & & -a_2 \\ & & 1 & \ddots & & \vdots \\ & & & \ddots & 0 & -a_{m-2} \\ & & & & 1 & -a_{m-1} \end{bmatrix}$$

Veja que a linha $(1, z, z^2, \ldots, z^{m-1})$ é um autovetor à esquerda de A, com autovalor z, quando z é uma raiz do polinônio $p(z)$.

$\Rightarrow$ Verifique que $(1, z, z^2, \ldots, z^{m-1})A = (1, z, z^2, \ldots, z^{m-1})z$, quando $p(z) = 0$

Teorema [Abel, 1824]

Para qualquer $m \geq 5$, existe um polinômio de grau m com coeficientes racionais que possui uma raiz real r, com a propriedade de que r não pode ser escrita por uma expressão envolvendo números racionais, adições, subtrações, multiplicações, divisões e radiciação.

Consequências:

1. Ainda que utilizássemos aritmética exata, não haveria algoritmo que produziria as raízes de um polinômio arbitrário em um número finito de passos.
2. A conclusão obviamente se aplica para o problema de se encontrar os autovalores de matrizes.

$\Rightarrow$ Qualquer algoritmo para o cálculo de autovalores deve ser iterativo e não baseado em algum método direto, como os que vimos para a solução de sistemas lineares.

Natureza iterativa dos algoritmos

- O objetivo dos algoritmos é produzir uma sequência de matrizes que rapidamente convirjam para uma forma que revele os autovalores.
- O cálculo de autovalores é portanto computacionalmente mais custoso que a solução de sistemas lineares.
- Em muitos casos, é possível produzir algoritmos que gerem sequências em que o número de digítos de precisão dobra ou triplica a cada iteração.

$\Rightarrow$ aplicar uma sequência de transformações unitárias $Q_j^* \cdots Q_2^* Q_1^* A Q_1 Q_2 \cdots Q_j$, de forma que

$$\lim_{j \to \infty} Q_j^* \cdots Q_2^* Q_1^* A Q_1 Q_2 \cdots Q_j = T$$

onde T é uma triangular superior.

Isto é, no limite, obteremos uma fatoração de Schur de A.

Obsevações:

1. Se A é real não simétrica, seus autovalores podem ser complexos (em conjugados). Portanto a fatoração deve admitir aritmética complexa.
2. Se A é Hermitiana, $Q_j^* \cdots Q_2^* Q_1^* A Q_1 Q_2 \cdots Q_j$ será hermitiana. Portanto T será triangular e hermitiana, ou seja, diagonal.

Independentemente de A ser simétrica ou não, os métodos para cálculo de autovalores se baseiam em duas fases:

- Fase I: método direto para transformar A em uma matriz Hessenberg superior:

  Isto é, uma matriz com zeros abaixo da primeira subdiagonal (quase uma triangular superior, exceto pela primeira subdiagonal). Custo $O(n^3)$. O objetivo desta fase é melhorar a convergência e o custo por iteração da Fase II.

- Fase II: método iterativo para, assintoticamente, transformar a Hessenberg superior em uma triangular superior.

  Cada iteração custa $O(n^2)$. Em princípio, "esta fase não termina nunca", mas com $O(n)$ iterações, a norma da subdiagonal inferior é reduzida para precisão da máquina. Sem a Fase I, o custo por iteração seria $O(n^3)$, pois a matriz seria densa.

Veja a estrutura das matrizes obtidas ao longo das duas fases quando:

- $A \neq A^*$ (não hermitiana)

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} \xrightarrow[\text{Fase 1}]{} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & & x & x \end{bmatrix} \xrightarrow[\text{Fase 2}]{} \begin{bmatrix} x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & & x & x \\ & & & & x \end{bmatrix}$$

- $A = A^*$.

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} \xrightarrow[\text{Fase 1}]{} \begin{bmatrix} x & x & & & \\ x & x & x & & \\ & x & x & x & \\ & & x & x & x \\ & & & x & x \end{bmatrix} \xrightarrow[\text{Fase 2}]{} \begin{bmatrix} x & & & & \\ & x & & & \\ & & x & & \\ & & & x & \\ & & & & x \end{bmatrix}$$

$\Rightarrow$ Premissa para qualquer ideia: usar transformações ortogonais, de forma a obter uma transformação similar.

- Primeira ideia: constuir um refletor de Householder Q_1^*, aplicar Q_1^* à esquerda de A, gerando zeros nas linhas $2, \ldots, m$ na primeira coluna e, depois, Q_1 à direita de $Q_1^* A$. Ao fazer esta segunda operação, combinaríamos as colunas de $Q_1^* A$, destruindo a estrutura de zeros criada na primeira coluna.

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} \xrightarrow{Q_1^* A} \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix} \xrightarrow{Q_1^* A Q_1} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix}$$

$\Rightarrow$ Não é surpresa que não possamos fazer esta transformação por uma sequência finita de passos, à luz do resultado de Abel.

$\Rightarrow$ Seremos menos ambiciosos na primeira fase: usaremos uma matriz ortogonal Q_1^* construída de forma que, na primeira iteração, mantenha inalterada a primeira linha, criando zeros a partir da terceira linha em diante.

- Ao construirmos esta "matriz mais modesta", não pederemos a estrutura da primeira coluna, quando fizermos a transformação à direita de $Q_1^* A$.

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix} \xrightarrow{Q_1^* A} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix} \xrightarrow{Q_1^* A Q_1} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & x & x & x & x \\ & x & x & x & x \end{bmatrix}$$

$\Rightarrow$ Repetimos a ideia nas colunas subsequentes.

- Ingrediente básico: Refletores de Householder de dimensões convenientes.

Continuamos com a mesma ideia nas colunas seguintes. No total, faremos $(n-2)$ transformações de Householder, à esquerda e à direita.

Na segunda iteração, a matriz Q_2^* preserva as duas primeiras linhas de $Q_1^* A Q_1$, criando zeros a partir da quarta linha....

$$\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & x & x & x & x \\ & x & x & x & x \end{bmatrix} \xrightarrow{Q_2^* Q_1^* A Q_1} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & 0 & x & x & x \\ & 0 & x & x & x \end{bmatrix} \xrightarrow{Q_2^* Q_1^* A Q_1 Q_2} \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & x & x & x \end{bmatrix}$$

A matriz H recebe uma cópia da matriz A.

- Na primeira coluna: Zeramos os elementos nas linhas $3, \ldots, m$. Portanto, para a construção de Q_1^*, empregamos:
  - $x \in \mathbb{R}^{m-1}$ como o vetor $H(2:m,1)$;
  - $v = signal(x_1)\|x\|e_1 + x$;
  - A matriz $Q_1^* \in \mathbb{R}^{m\times m}$ será $Q_1^* = \begin{bmatrix} 1 & 0_{m-1}^T \\ 0_{m-1} & F \end{bmatrix}$, obtida a partir do refletor $F = I - \frac{2}{\|v\|^2} vv^T \in \mathbb{R}^{(m-1)\times(m-1)}$.
- Para qualquer coluna, de índice $k \in \{2, \ldots, m-2\}$
  - $x \in \mathbb{R}^{m-k}$ é o vetor $H(k+1:m,k)^T$;
  - $v = signal(x_1)\|x\|e_1 + x$;
  - A matriz $Q_k^* \in \mathbb{R}^{m\times m}$ será $Q_k^* = \begin{bmatrix} I_k & 0_{m-k}^T \\ 0_{m-k} & F \end{bmatrix}$, obtida a partir do refletor $F = I - \frac{2}{\|v\|^2} vv^T \in \mathbb{R}^{(m-k)\times(m-k)}$.

```
function [H] = Householder_To_Hessenberg(A)
    [m,n] = size(A)
    if (m <> n) then
        printf('Matriz não é quadrada. \n')
    end
    H = A
    for k = 1:m-2
        x = H(k+1:m,k)

        vk = sign(x(1))*norm(x,2) * eye(m-k,1) + x
        vk = 1.0 / norm(vk,2) * vk

        // multiplicacao a esquerda, operacoes em linhas de H
        H(k+1:m,k:m)   = H(k+1:m,k:m)   - 2.0 * vk*(vk'*H(k+1:m,k:m))

        // multiplicacao a direita, operacoes em colunas de H
        H(1:m,k+1:m)   = H(1:m,k+1:m)   - 2.0 * (H(1:m,k+1:m) * vk) * vk'
    end
endfunction
```

A matriz de entrada é simétrica. Resultado: Tridiagonal.

```
A1  =
   85.     102.    70.     129.    137.
   102.    167.    85.     157.    189.
   70.     85.     110.    91.     151.
   129.    157.    91.     272.    218.
   137.    189.    151.    218.    267.


->H =  Householder_To_Hessenberg(A1)


 H  =
   85.         -225.19769    1.701D-14    3.283D-14    2.122D-14
  -225.19769    683.96883    7.3768307   -6.085D-14   -8.860D-14
  -1.421D-14    7.3768307    72.467708   -28.312681    3.553D-14
  -2.842D-14    8.882D-16   -28.312681    30.104996    23.589342
  -2.842D-14   -1.776D-15    1.066D-14    23.589342    29.458471
```

A matriz de entrada não é simétrica (neste exemplo particular, admite autovalores reais). Resultado: Hessenberg Superior.

```
A2  =
   10.    4.    9.    8.    2.
   8.     1.    8.    7.    7.
   5.     3.    6.    1.    9.
   7.     4.    4.    3.    5.
   0.     2.    9.    7.    3.


->H =  Householder_To_Hessenberg(A2)


 H  =
   10.        -11.321712  -1.8717159   4.2299349  -3.9272345
  -11.74734    11.536232   11.539886  -5.3539833   3.8128417
   0.          9.3998608   3.0516512  -5.2453526   4.851592
   0.         -1.110D-16  -4.6624397   1.7493343   1.8270586
   0.          1.776D-15  -8.882D-16   1.2442563  -3.3372174
```

O algoritmo a seguir pode ser usado sem a chamada prévia da Fase I, porém precisará de mais iterações para convergir.

Seja A a matriz recebida. O que o algoritmo na Fase II retorna:

- Se A é Hessenberg (triangular superior + subdiagonal), o algoritmo irá produzir uma fatoração de Schur de A. O espectro de A estará representado na diagonal.
- Se A é tridiagonal, o algoritmo irá produzir uma fatoração espectral para A (ou uma diagonalização de A), isto é, teremos a matriz de autovetores e uma diagonal com seus autovalores.

1. Fatoramos $A = QR$.
2. Multiplicamos A pelo fator Q^T, à esquerda, e Q à direita, obtendo uma transformação similar:

$$\begin{aligned} A &= QR \\ Q^T AQ &= Q^T QRQ \\ &= RQ \end{aligned}$$

3. Atualizamos A como RQ e repetimos os passos 1 a 2, "até convergir".
4. O algoritmo converge para uma A triangular superior, com os autovalores na diagonal.

Observações importantes:

1. Este algoritmo essencialmente explora a primeira ideia descartada para a triangularização de A na Fase I.

2. Esta ideia é ruim para transformar A em uma triangular superior em um único passo, mas é bastante eficiente como estrutura de um processo iterativo, que gera uma forma de Schur para A, principalmente se a Fase I tiver sido chamada previamente.

Vamos usar fatorações QR, via Householder.

Algoritmo comentado:

(Inicialização:) $k \leftarrow 1$, $A^{(k)} = A$.

Repita até convergir:

1. Fatoramos $A^{(k)} = Q^{(k)} R^{(k)}$ (Householder faz a triangularização de $A^{(k)}$), isto é, $R^{(k)} = (Q^{(k)})^T A^{(k)}$).
2. Como $Q^{(k)}$ é ortogonal, $(Q^{(k)})^T A^k = R^{(k)}$.
3. Multiplicando à direita: $(Q^{(k)})^T A^k Q^{(k)} = R^{(k)} Q^{(k)}$ (temos uma transformação similar, que preserva o espectro). Esta operação destrói a estrutura triangular de $R^{(k)}$....
4. $A^{(k+1)} \leftarrow R^{(k)} Q^{(k)}$, $k \leftarrow k+1$ e repetimos o processo, até após a atualização, $A^{(k+1)}$ seja suficientemente triangular superior.

Algoritmo QR para decomposição espectral

```
function [Qa,Lambda,H] = QR_Para_Autovalores(A,tol)
    [m,n] = size(A)
    H = A
    [H] = Householder_To_Hessenberg(A)
    CONVERGIU = 0
    k = 0
    Qa = eye(m,m)
    while (CONVERGIU == 0)
       [Q,R] = QR_Householder(H)
       Qa = Qa * Q
       H = R*Q
       ninf = norm(tril(H)-diag(diag(H,0)),1)
       printf("Iter: %d ninf: %7.6E \n",k,ninf)
       k = k + 1
       if (ninf < tol)
         CONVERGIU = 1
       end
       k = k+1
    end
    Lambda = diag(H)
endfunction
```

Exemplo I

```
A1  =
   85.     102.    70.     129.    137.
   102.    167.    85.     157.    189.
   70.     85.     110.    91.     151.
   129.    157.    91.     272.    218.
   137.    189.    151.    218.    267.

[Qa1,Lambda1,T1] = QR_Para_Autovalores(A1,1.0E-15)

Lambda1  = 759.26225 88.740522 43.220379 9.7447831 0.0320717

T1  =

   759.26225  -2.324D-14   6.974D-15  -5.490D-14   9.601D-14
   0.          88.740522   1.030D-14   1.285D-14   1.225D-14
   0.          0.          43.220379   8.313D-15  -1.465D-14
   0.          0.          0.          9.7447831   3.233D-15
   0.          0.          0.          0.          0.0320717
```

Exemplo II

```
A2  =

   10.    4.    9.    8.    2.
   8.     1.    8.    7.    7.
   5.     3.    6.    1.    9.
   7.     4.    4.    3.    5.
   0.     2.    9.    7.    3.

[Qa2,Lambda2,T2] = QR_Para_Autovalores(A2,1.0E-15)

Lambda2  = 26.024819 -7.5953848 6.7069158 -3.3870332 1.2506831

 T2  =

   26.024819   0.8270619   4.8318138   8.2921898  -2.0623582
   0.         -7.5953848   0.7415849  -2.272945   -1.8566947
   0.          0.          6.7069158   0.6733713   1.4334871
   0.          0.          0.         -3.3870332  -0.5328745
   0.          0.          0.          0.          1.2506831
```

Discutimos a abordagem inocente:

1. Calculamos explicitamente $A^T A$.
2. Fatoramos $A^T A = Q^T \Lambda Q$.
3. Os valores singulares σ_i de A são as raízes dos autovalores positivos λ_i de $A^T A$, armazenados na diagonal de Λ.
4. Os vetores singulares à direita v_i de A são as colunas q_i de Q, associadas às entradas $\lambda_i > 0$.
5. Os vetores singulares à esqueda de A são obtidos via $u_i = \frac{1}{\sigma_i} A q_i$, para todo $\sigma_i > 0$.

$\Rightarrow$ "Perda de informação devido ao quadrado". Devemos evitar esta abordagem pois $A^T A$ é usualmente mal condicionada: Se A possui valores singulares distintos de zero mas muito pequenos, estes valores serão avaliados sem precisão.

Os algoritmos que vamos discutir operam diretamente em A (e não em $A^T A$ ou AA^T).

Estrutura dos algoritmos:

- Fase I: Transformamos A em uma matriz bidiagonal $\hat{B}$ por meio de um método direto (baseado em transformações ortogonalmente equivalentes, via Householder).
- Fase II: Extraímos de $\hat{B}$ sua submatriz B bidiagonal quadrada.
  - Construímos uma matriz auxiliar quadrada $(2n \times 2n)$ $H = \begin{bmatrix} 0 & B^* \\ B & 0 \end{bmatrix}$.
  - Fazemos a fatoração espectral de $H = Z\Sigma Z^T$. Com os autovetores de H (nas colunas de Z) recuperamos os vetores singulares de $B, \hat{B}$ e de A.

Duas matrizes $\hat{B}, A \in \mathbb{R}^{m \times n}$ são ortogonalmente equivalentes (OE) se e somente se existem matrizes ortogonais $Q \in \mathbb{R}^{m \times m}, P \in \mathbb{R}^{n \times n}$ tais que

$$A = Q\hat{B}P$$

- Seja $\hat{B} = \hat{U}\Sigma\hat{V}^T$ a fatoração SVD de $\hat{B}$.
- Então para $A, \hat{B}$ OE satisfazendo $\hat{B} = Q^T A P^T$:

$$\begin{aligned} A &= Q\hat{B}P \\ &= Q(\hat{U}\Sigma\hat{V}^T)P \\ &= (Q\hat{U})\Sigma(\hat{V}^T P) \end{aligned}$$

$$\hat{B} = \hat{U}\Sigma\hat{V}^T \iff A = (Q\hat{U})\Sigma(\hat{V}^T P)$$

Consequências:

- $A, \hat{B}$ possuem os mesmos valores singulares σ_i.
- Os vetores singulares à esquerda e à direita de A são dados respectivamente pelas colunas de $Q\hat{U}$ e $\hat{V}^T P$ associados às entradas $\sigma_i > 0$.

$$\hat{B} = \hat{U}\Sigma\hat{V}^T \iff A = (Q\hat{U})\Sigma(\hat{V}^T P)$$

Observações:

1. No caso da fatoração SVD não precisamos de transformações similares. Precisamos de "menos do que isso": transformações ortogonalmente equivalentes.
2. Vamos transformar A em uma $\hat{B}$ conveniente, ortogonalmente equivalente a A, e calcular a fatoração SVD de $\hat{B}$.
3. No contexto de fatoração SVD, a forma "conveniente"da matriz $\hat{B}$ é bidiagonal.

$\Rightarrow$ Assumimos de agora em diante que $m \geq n$ (caso contrário fazemos a SVD de A^T).

Podemos bidiagonalizar $A \in \mathbb{R}^{m \times n}$, por meio de n matrizes ortogonais $E_i \in \mathbb{R}^{m \times m} : i = 1, \ldots, n$ e $n-2$ matrizes ortogonais $D_i \in \mathbb{R}^{n \times n} : i = 1, \ldots, n-2$, aplicados sequencialmente à esquerda e à direita de A, respectivamente.

Isto é, existem $O(n)$ matrizes E_i, D_i ortogonais tais que:

$$\hat{B} = E_n E_{n-1} \ldots E_1 A D_1 D_2 \ldots D_{n-2}$$

onde $\hat{B}$ é bidiagonal, ortogonalmente equivalente a A.

As matrizes E_i e D_i serão construídas por meio de refletores de Householder.

- Vamos supor que a transformação E_1A na primeira coluna de A tenha sido realizada: $E_1A = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 3 & -2 \\ 0 & 2 & 4 & 1 \\ 0 & -3 & -2 & 5 \end{bmatrix}$
- $D_1 = \begin{bmatrix} 1 & 0 \\ 0 & F_1 \end{bmatrix}$ onde $F_1 = I_{3\times 3} - 2\frac{vv^T}{v^Tv}$ para $x = \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T$, $v = \sqrt{2}\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T - \begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T = \begin{bmatrix} \sqrt{2}-1 & -1 & 0 \end{bmatrix}^T$
- $D_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
- $E_1AD_1 = \begin{bmatrix} 1 & 1.4142136 & 0 & 0 \\ 0 & 2.8284271 & -1.4142136 & -2 \\ 0 & 4.2426407 & -1.4142136 & 1 \\ 0 & -3.5355339 & -0.7071068 & 5 \end{bmatrix}$

```
function [B] = Bidiagonaliza(A)
    [m,n] = size(A)
    if (n > m) then B = A'; [m,n] = size(B);
               else B = A;
    end
    for k = 1:n
        // operacoes a esquerda
        x = B(k:m,k)
        vk = sign(x(1))*norm(x,2) * eye(m-k+1,1) + x
        vk = 1.0 / norm(vk,2) * vk
        B(k:m,k:n)   = B(k:m,k:n)   - 2.0 * vk*(vk'*B(k:m,k:n))
        if (k <= n-2)
          // Operacoes a direita
          x  = B(k,k+1:n)'
          [n1,m1] = size(x)
          vk = sign(x(1))*norm(x,2) * eye(n-k,1) + x
          vk = 1.0 / norm(vk,2) * vk
          D = eye(n,n)
          F = eye(n-k,n-k)
          F = F - 2 * vk * vk'
          D(k+1:n,k+1:n) = F
          B = B * D
        end
    end
endfunction
```

```
-->Bidiagonaliza(A1)
 ans  =

  -240.70521    719.46083    0.           0.           0.
   0.           31.902753    22.076149    0.           0.
   0.           0.           83.604679   -17.802383    0.
   0.           0.           0.           43.573255    6.6092123
   0.           0.           0.           0.          -0.0325336

-->Bidiagonaliza(A2)
 ans  =

  -15.427249    20.884335    0.           0.           0.
   0.           12.153633    2.4112356    0.           0.
   0.           0.          -4.5019751   -5.7067634    0.
   0.           0.           0.           3.7099871    2.1224413
   0.           0.           0.           0.           1.7933167
```

Fase I: Exemplo

```
->A
 A3  =

   2.    4.    5.    8.
   6.    9.    3.    0.
   6.    9.    5.    7.
   7.    3.    5.    2.
   9.    4.    1.    4.
   5.    7.    2.    8.
   3.    3.    6.    6.

-->[B3] = Bidiagonaliza(A3)
 B3  =
  -15.491933   19.530425   0.          0.
   0.         -12.286121  -3.5853615   0.
   0.           0.        -6.8825697  -0.431447
   0.           0.         0.          5.1188868
   0.           0.         0.          0.
   0.           0.         0.          0.
   0.           0.         0.          0.
```

- Assumimos que a matriz bidiagonal B $(n \times n)$ obtida (extraída de $\hat{B}$) é *propriamente bidiagonal*, ou seja, não há elementos nulos na diagonal principal e na primeira super diagonal.

- Suponha o contrário:

  - Se $B_{k,k+1} = 0$ para algum k, podemos particionar $B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}$ em blocos, onde $B_1 \in \mathbb{R}^{k \times k}$, $B_2 \in \mathbb{R}^{m-k \times m-k}$. As fatorações SVD de B_1, B_2 podem ser feitas separadamente e, então, combinadas para a SVD de B.
  - Se $B_{k,k} = 0$ também podemos decompor o problema em dois subproblemas independentes (detalhes omitidos aqui).

Para B propriamente bidiagonal, a matriz auxiliar H pode ser escrita como

$$\begin{bmatrix} 0 & B^* \\ B & 0 \end{bmatrix} \begin{bmatrix} V & V \\ U & -U \end{bmatrix} = \begin{bmatrix} V & V \\ U & -U \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & -\Sigma \end{bmatrix}$$

Observações:

1. H é não singular, tendo todos seus autovalores reais não nulos.
2. Os autovalores de H aparecem aos pares $\sigma_i, -\sigma_i$. Os módulos destes valores fornecem os valores sigulares de B.
3. Se $\begin{bmatrix} v & u \end{bmatrix}^T \in \mathbb{R}^{2n}$ é autovetor de H, $\begin{bmatrix} v & -u \end{bmatrix}^T$ também é. u, v são vetores singulares à direita e à esquerda de B. Os vetores v_i são vetores singulares de B e de $\hat{B}$.
4. Os vetores singulares de A podem ser computados a partir de v, u.

Fase II: Exemplo (continua)

```
->A3
 A3  =
   2.   4.   5.   8.
   6.   9.   3.   0.
   6.   9.   5.   7.
   7.   3.   5.   2.
   9.   4.   1.   4.
   5.   7.   2.   8.
   3.   3.   6.   6.

-->[B3] = Bidiagonaliza(A3);

Bp = B3(1:4,1:4)
H = zeros(8,8);
H(1:4,5:8) = Bp';
H(5:8,1:4) = Bp;
```

Fase II: Exemplo (continua)

```
->[Q,S] = spec(H);
->diag(S)'
 ans  =
  -26.913271  -8.8292206  -5.5837141  -5.0539917   5.0539917
    5.5837141  8.8292206   26.913271

-->[U,S,V] = svd(Bp);
-->diag(S)'
 ans  =
   26.913271   8.8292206   5.5837141   5.0539917
```

Facilitando o cálculo dos vetores singulares

1. A exposição anterior se concentrou no cálculo dos valores singulares de uma matriz retangular $A \in \mathbb{R}^{m \times n}$.
2. Vamos simplificar os passos anteriores para também dispormos dos vetores singulares.
3. Os procedimentos apresentados até agora serão particularizados para o caso de uma matriz quadrada.
4. Para isso, vamos fatorar a matriz A em $A = QR$, fazer a fatoração SVD de R (que é quadrada) e compor o resultado.

Desejamos obter os fatores U, Σ, V em $A = U\Sigma V^T$.

P.1 Dada $A \in \mathbb{R}^{m\times n}$, fazemos a fatoração $A = QR$ reduzida de A. Assumimos que A possui posto coluna completo.

P.2 Fatoramos $R \in \mathbb{R}^{n\times n}$ na forma *SVD* obtendo:

$$R = U_R \Sigma_R V_R^T$$

P.3 Compomos o resultado:

$$A = (QU_R)\Sigma_R V_R^T$$

$$\Rightarrow \text{Ou seja: } V = V_R, U = QU_R, \Sigma = \Sigma_R.$$

Detalhamento do Passo 2:

P. 2.1 Bidiagonalização de R por meio de transformações unitárias:

$$ERD = B \iff R = E^T B D^T$$

P. 2.2 Fatoração espectral de $H = \begin{bmatrix} 0 & B^* \\ B & 0 \end{bmatrix}$. Recordando:

$$\begin{bmatrix} 0 & B^* \\ B & 0 \end{bmatrix} \begin{bmatrix} V_B & V_B \\ U_B & -U_B \end{bmatrix} = \begin{bmatrix} V_B & V_B \\ U_B & -U_B \end{bmatrix} \begin{bmatrix} \Sigma_B & 0 \\ 0 & -\Sigma_B \end{bmatrix}$$

Ou seja,

$$BV_B = U_B \Sigma_B,$$

e portanto, a fatoração espectral de H nos fornece a SVD de B (e vice versa).

Atenção: Para uso na fatoração de A, precisamos normalizar V_B, U_B. Isso porque extraímos V_B, U_B de um autovetor $z = [V_B \quad U_B]^T$ tem $\|z\|_2 = 1$. Porém, seus subvetores V_B, U_B não.

1. Fatoração SVD de B:

$$B = U_B \Sigma_B V_B^T$$

2. Fatoração SVD de R:

$$\begin{aligned} R &= E^T B D^T \\ &= (E^T U_B) \Sigma_B V_B^T D^T \end{aligned}$$

3. Fatoração QR de A:

$$\begin{aligned} A &= QR \\ &= (QE^T U_B) \Sigma_B (V_B^T D^T) \end{aligned}$$

4. Fatores obtidos em $A = U \Sigma V^T$:

$$U = QE^T U_B,\ \Sigma = \Sigma_R,\ V^T = V_B^T D^T$$

Usando os algoritmos disponíveis no Scilab

```
A3 = [2.   4.   5.   8.; 6.   9.   3.   0.;
      6.   9.   5.   7.;  7.   3.   5.   2.;
      9.   4.   1.   4.;  5.   7.   2.   8.; 3.   3.   6.   6.];

-->[Us,Ss,Vs] = svd(A3);

-->Vs,Ss

 Vs  =
  -0.528988   -0.568151   -0.5957387   0.2060861
  -0.5710213  -0.2856276   0.7690584  -0.0300126
  -0.3702612   0.2547129  -0.214162   -0.8672731
  -0.5069645   0.7285209  -0.0881998   0.4521782

 Ss  =
   26.913271   0.          0.          0.
   0.          8.8292206   0.          0.
   0.          0.          5.5837141   0.
   0.          0.          0.          5.0539917
   0.          0.          0.          0.
   0.          0.          0.          0.
   0.          0.          0.          0.
```

A nossa formulação para resolver o problema:

```
[m,n] = size(A3);
[Q,R] = qr(A3);
[E,B,D] = BidiagonalizaExp(R(1:n,1:n))
H = zeros(2*n,2*n);
BTrans = B';
H(1:n,n+1:2*n) = BTrans;
H(n+1:2*n,1:n) = B;
[AutovetoresH,S] = spec(H)
VB = AutovetoresH(1:n,n+1:2*n)
UB = AutovetoresH(n+1:2*n,n+1:2*n)
for i = 1:n
    VB(:,i) = VB(:,i)/norm(VB(:,i),2)
    UB(:,i) = UB(:,i)/norm(UB(:,i),2)
end
U = Q(:,1:n)*E'*UB
V = D*VB
Sigma = diag(S(n+1:2*n,n+1:2*n))
```

Nosso resultado:

```
-->V

 V  =

   0.2060861  -0.5957387  -0.568151   -0.528988
  -0.0300126   0.7690584  -0.2856276  -0.5710213
  -0.8672731  -0.214162    0.2547129  -0.3702612
   0.4521782  -0.0881998   0.7285209  -0.5069645

-->Sigma

 Sigma  =

   5.0539917
   5.5837141
   8.8292206
   26.913271
```

Nosso resultado:

```
-->U

 U  =

  -0.0844516   0.0194042   0.5462452  -0.3436625
  -0.3235889   0.4843742  -0.5906995  -0.3501582
  -0.0405055   0.2970933   0.0445856  -0.5095322
  -0.4114458  -0.5570138  -0.2382241  -0.3077001
   0.5295151  -0.5108385  -0.3496427  -0.3508714
   0.5348687   0.2275891   0.169601   -0.425007
  -0.388277   -0.2317834   0.3780704  -0.3181844
```